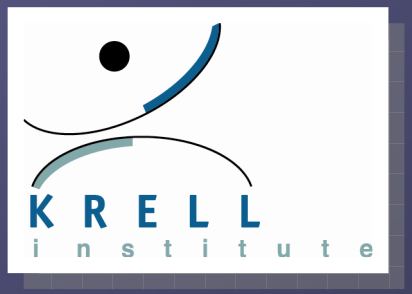
UCRL-PRES-230047

Open | SpeedShop™

Update on Open|SpeedShop: An Open-Source Performance Toolset for Linux Clusters

Martin Schulz
Lawrence Livermore National Laboratory



What is Open|SpeedShop?

- **Comprehensive Open Source Parallel Performance Analysis Framework**
  - Goal: ***One*** tool for all performance analysis needs
  - Targets Users ***and*** Tool Developers
- **Funding**
  - DOE/NNSA as part of ASC PathForward
  - Initial phase co-funded by SGI
- **Status**
  - Version 1.0 available as source and RPMs
  - Source code is GPL/LGPL

Partners

- **Krell Institute**
  - Hosts Development
- **DOE/NNSA Tri-Labs**
  - Lawrence Livermore
  - Los Alamos
  - Sandia
- **University of Wisconsin & University of Maryland**
  - DynInst & Infrastructure

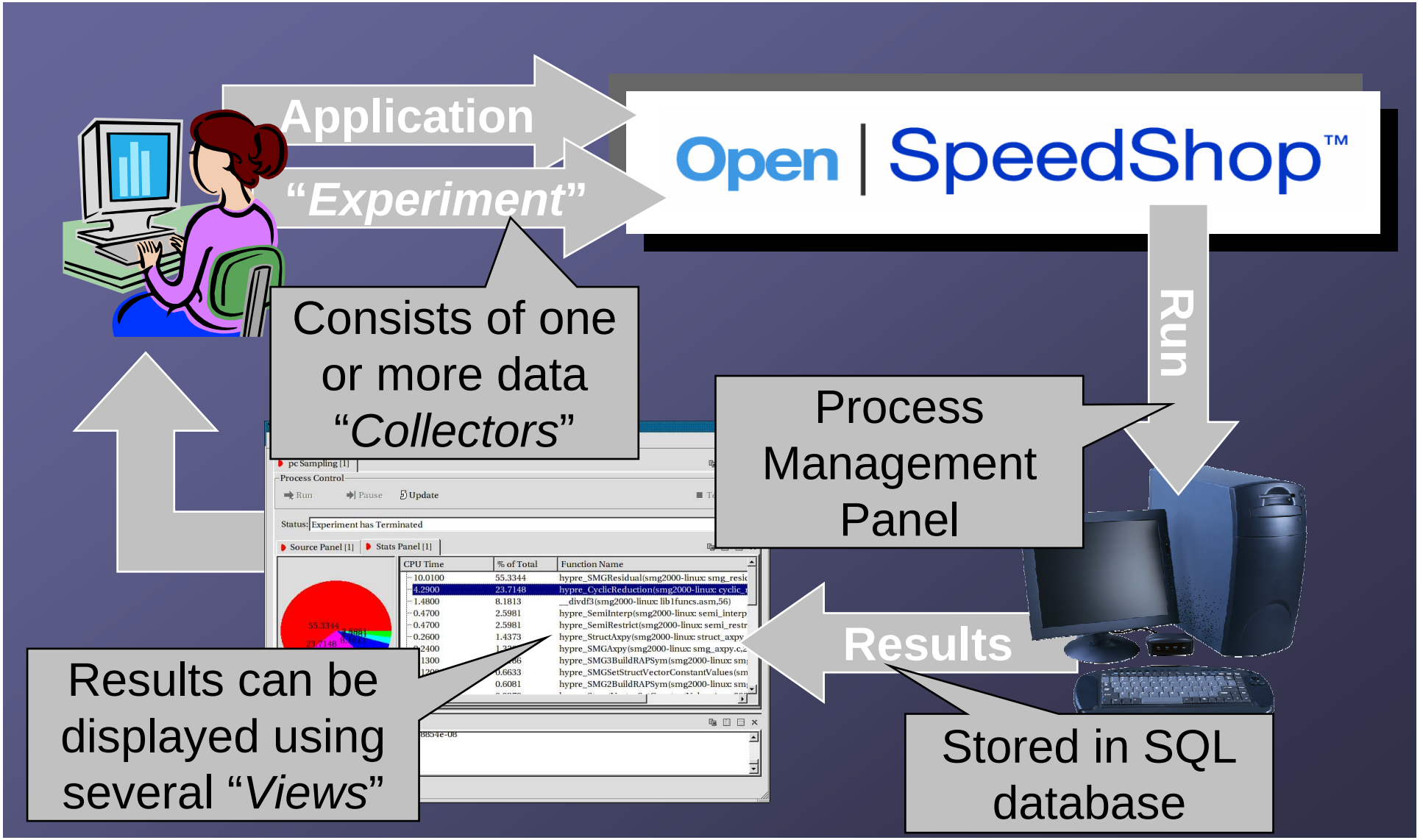
Highlights

- **Open Source Performance Analysis Tool Framework**
  - Most common performance analysis steps ***all in one tool***
  - ***Extensible*** by using plugins for data collection and representation
- **Instrumentation at Runtime**
  - Use of ***unmodified application binaries***
  - ***Attach*** to running applications
- **Flexible and Easy to use**
  - User access through ***GUI***, ***Command Line***, and ***Python Scripting***
- **Large Range of Platforms**
  - ***Linux Clusters*** with x86, IA-64, Opteron, and EM64T CPUs
  - Designed with ***portability*** and ***scalability*** in mind
- **Availability**
  - Used at ***all three ASC labs*** with lab-size applications
  - Source and RPM versions available

Outline

- **Workflow**
- **O|SS for Users**
  - "Hello World" Example
  - Available Features
- **O|SS for Developers**
- **Status & Roadmap**
- **Contact & Availability**

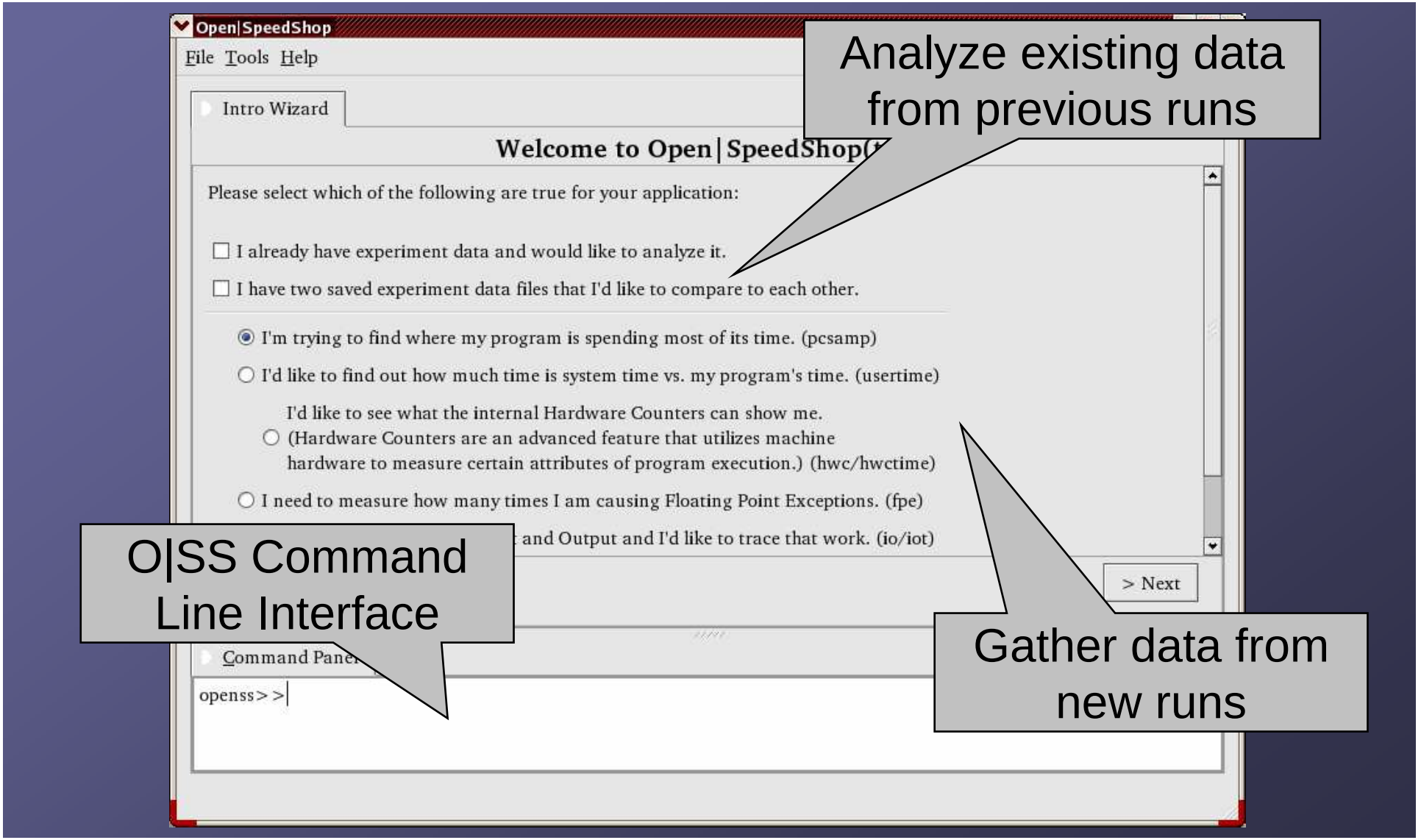
Workflow & Terminology



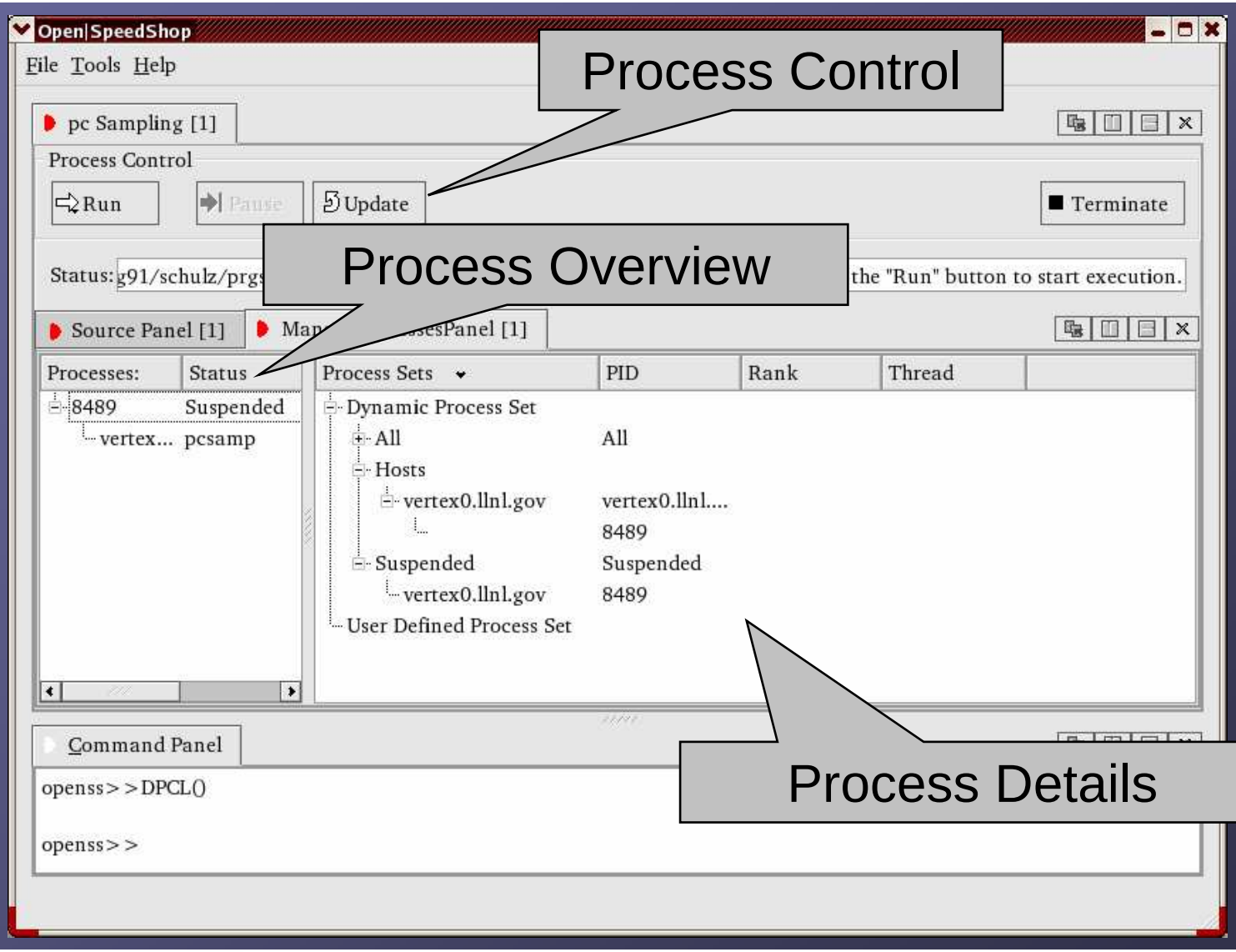
Performance Experiments

- **Define what data is collected and how**
- **Sampling**
  - PC sampling
  - User time (incl. stack information)
  - Hardware counter
- **Tracing**
  - MPI call tracing
  - I/O call tracing
  - Floating Point Exception (FPE) tracing

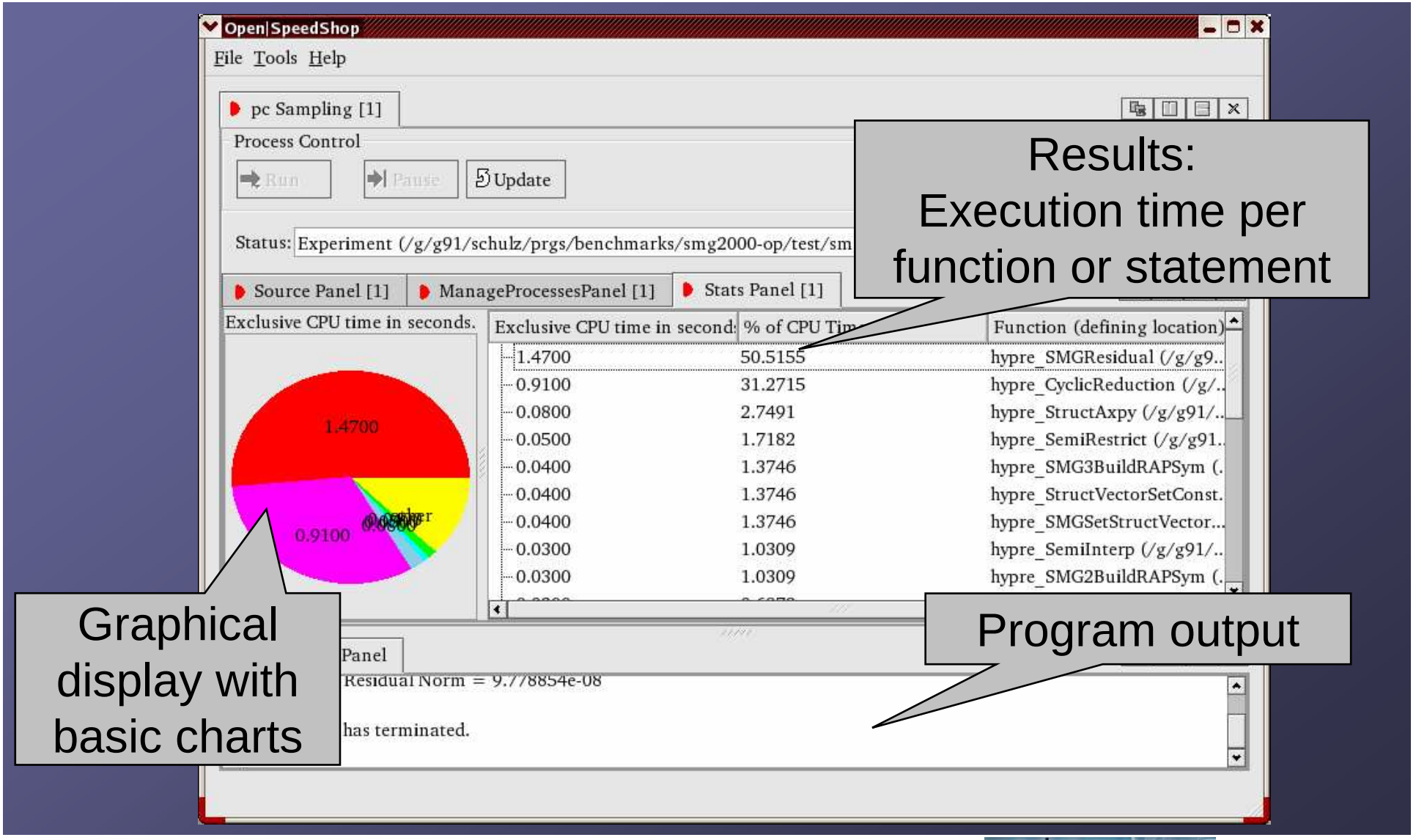
Opening screen



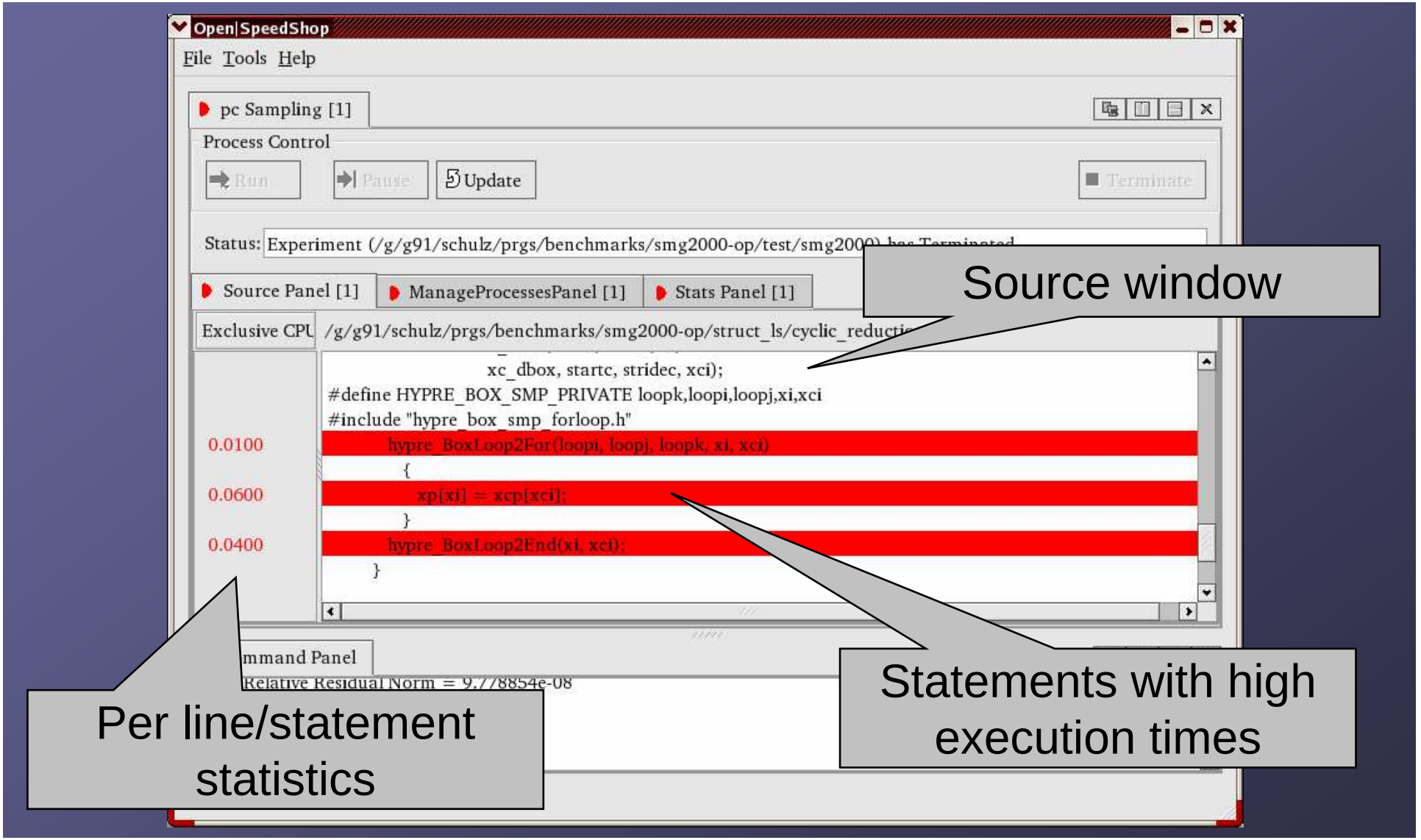
Process Management



PC Sampling Results



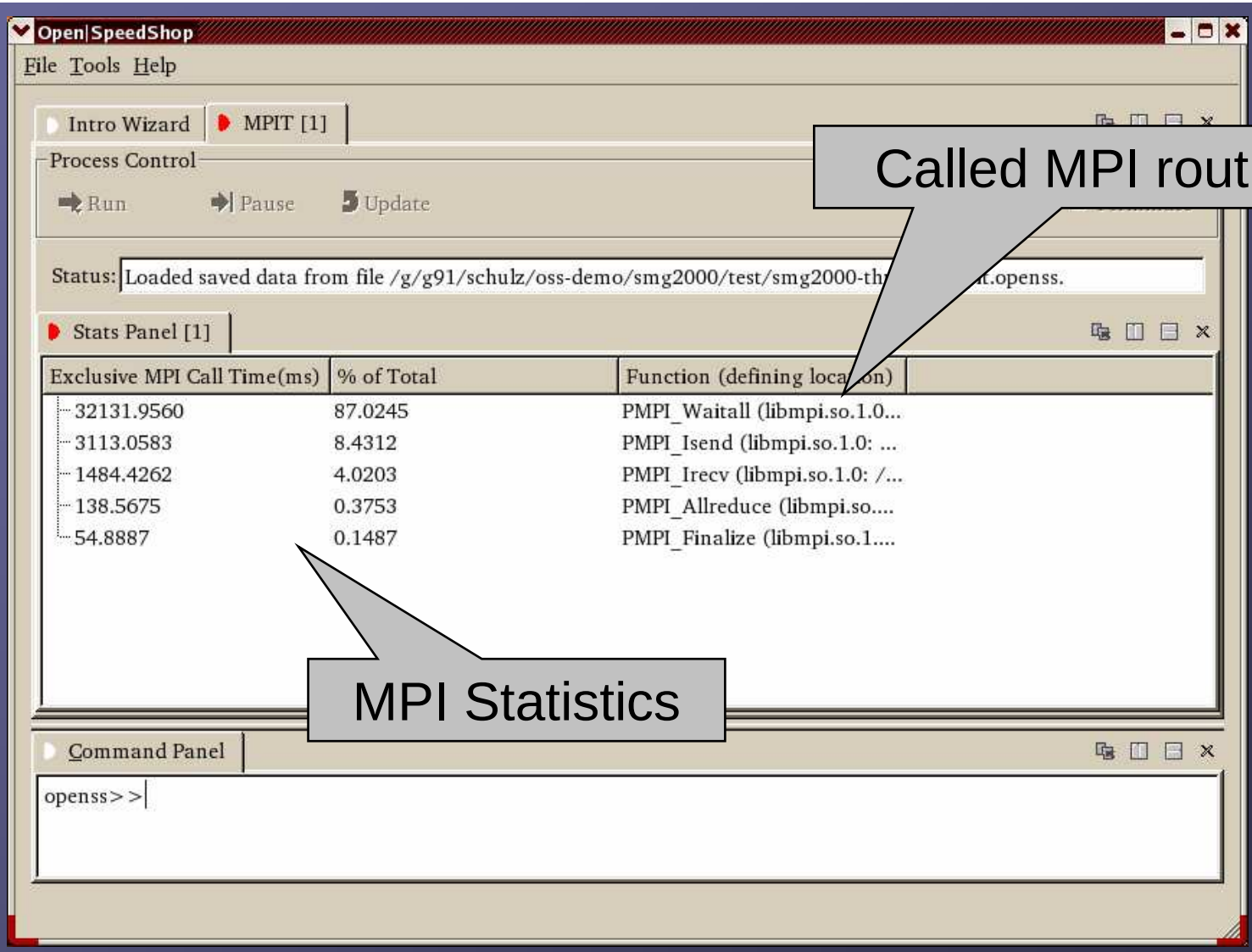
Line-by-Line Analysis



Parallel Performance Analysis

- **O|SS supports MPI and Multithreading**
- **Sequential experiments**
  - Apply experiment/collectors to all nodes
  - By default display aggregate results
  - Optional select individual groups of processes
- **MPI experiments**
  - Tracing of MPI calls
  - Can be combined with sequential collectors
- **Uses MPIR debug interface to query job**

Example: MPI Tracing



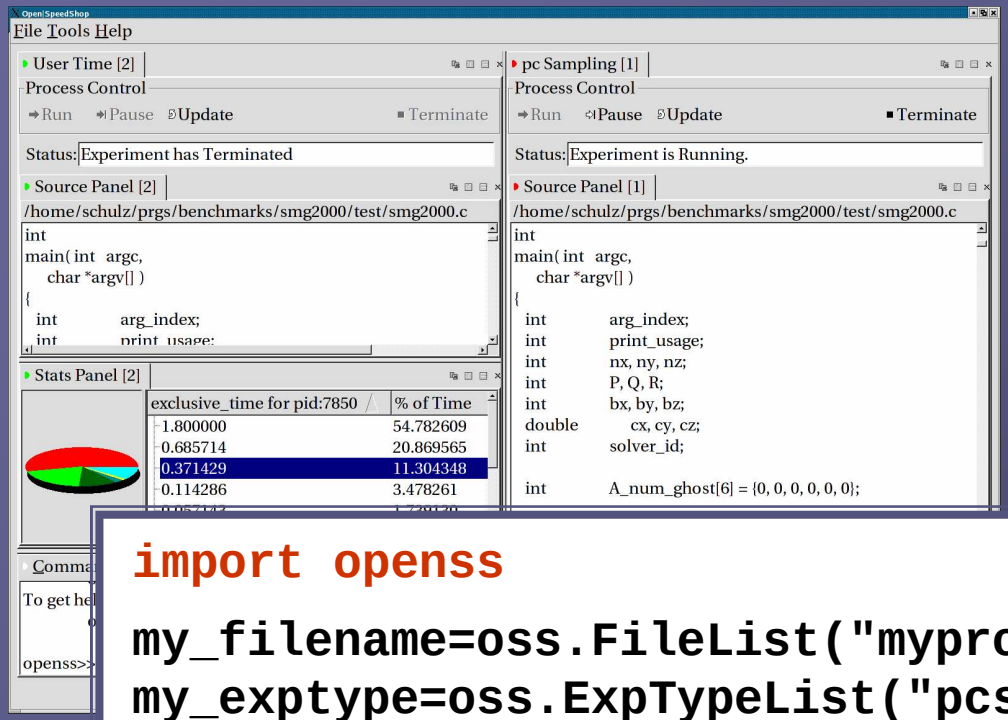
Advanced Capabilities

- **Stacktraces**
  - Included in tracing and user time experiments
  - Visualize as call-tree and trace-back
- **Comparisons**
- **Time segments**
- **Multi-rank analysis**
  - Restrict results to task sets
  - Compare tasks or task sets
  - Cluster Analysis

Overhead (on IA-64/Thunder)

- **Depends on Experiment and Application**
- **Sampling**
  - Adjustable parameter: sampling rate
  - PC Sampling: typically around 2%
  - HWC: sampling rate must match application
  - Example L2 cache misses: < 3%
- **Tracing**
  - Capture each event / can't be adjusted
  - Have seen overheads of 30-40%

User Interfaces



Experiment Commands

- expAttach
- expCreate
- expDetach
- expGo
- expView

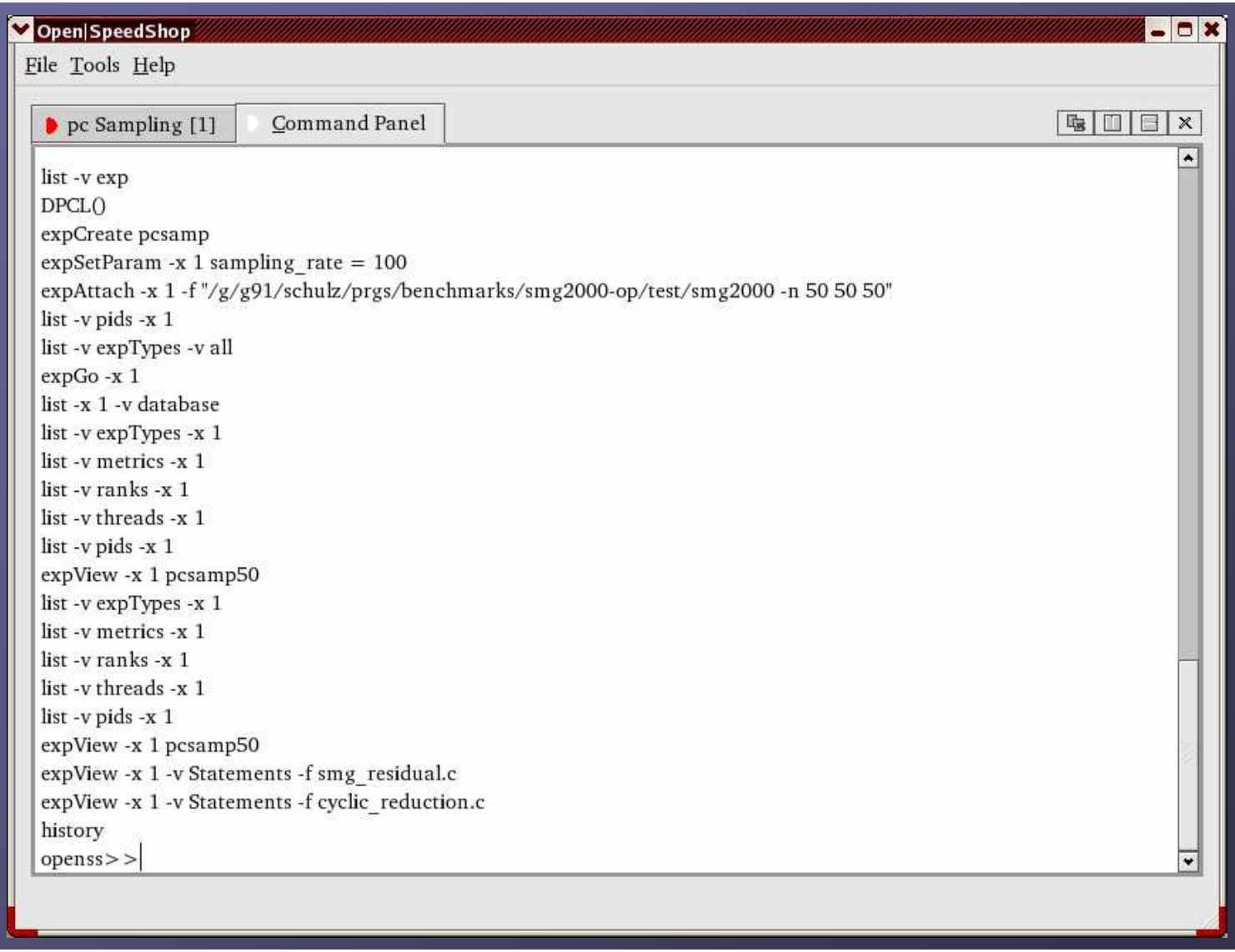
List Commands

```
import openss
my_filename=oss.FileList("myprog.a.out")
my_exptype=oss.ExpTypeList("pcsamp")
my_id=oss.expCreate(my_filename,my_exptype)

oss.expGo()

My_metric_list = oss.MetricList("exclusive")
my_viewtype = oss.ViewTypeList("pcsamp“)
result = oss.expView(my_id,my_viewtype,my_metric_list)
```

UI Implementation Details

- **All three interfaces built on top of CLI**
  - GUI issues equivalent CLI commands
  - Python commands translated into CLI calls
- **UIs can share state**
  - Switch from shell to GUI and back
  - Maintain open experiments
- **Advantages**
  - Guarantees equal functionality
  - Single path to test and debug
  - Common log and replay mechanism

History Command



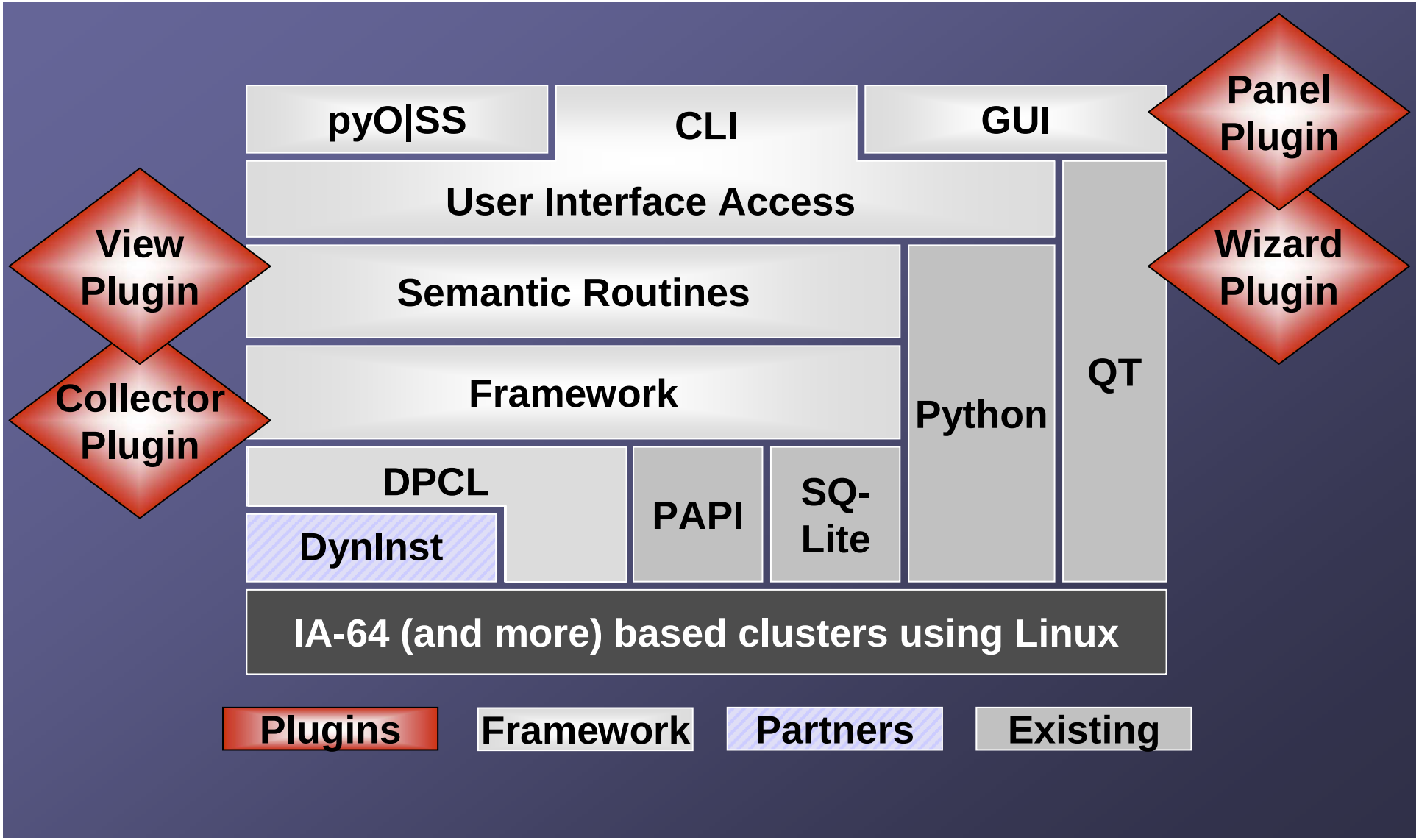
O|SS Target Audience

- **Programmers/code teams**
  - Use Open|SpeedShop out of the box
  - Powerful performance analysis
  - Works on unmodified binaries
  - Ability to integrate O|SS into projects
- **Tool developers (academia & industry)**
  - Single, comprehensive infrastructure
  - Same “look and feel” across all tools
  - Easy deployment of new tools
  - Commercial plugins possible

Adding Functionality

- **Plugins to add new tools**
  - Encapsulate only tool's core functionality
  - Inherit O|SS's look and feel
- **Advantages**
  - No need to re-implement infrastructure
  - Automatic support for parallel applications
  - Results can be compared with other tools
- **Easy deployment**
  - Install in global tool directory
  - No new software to install/learn

O|SS Architecture



Status

- **Version 1.0**
  - Released last November (SC07)
  - All six experiments
- **Available from sourceforge**
  - Binary and Source Versions
  - www.openspeedshop.org
- **Tutorials**
  - Available from Open|SpeedShop website
  - SC 2006 & ICS 2007

What's next?

- **Target: Peta-Scale machines**
  - Data Collection and Transport
  - Result storage, aggregation, and analysis
- **Offline Collectors**
  - Execute experiments without tool backend
  - Targets microkernel architectures
- **Fully disconnect GUI from framework**
  - Remote execution with local GUI
  - Built on Command Line Interface (CLI)
- **Long Term Vision**
  - Performance "Cookbooks"
  - Help users plan experiments

Building a Community

- **More then yet another research tool**
  - Large scale environment
  - Flexible framework
  - Cooperative atmosphere
- **Wanted:**
  - Early users
  - Open source tool developers
  - Tools/Software companies
- **Goal: self-sustaining project**

Summary

- **Open|SpeedShop**
  - "One-stop performance analysis"
  - Works on unmodified/running applications
- **Support for wide range of experiments**
  - Sampling (timing and hardware counters)
  - Tracing (MPI, I/O, FPE)
- **Easy and flexible user access**
  - GUI with Wizards
  - Scripting and batch processing
- **Plugin infrastructure to extend functionality**

Availability and Contact

- **Open|SpeedShop website:**
  http://www.openspeedshop.org/
- **Download options:**
  - RPMs for several platforms
  - Source for tool and base libraries
- **Feedback**
  - Bug tracking available from website
  - Contact information on website
  - Email: oss-questions@openspeedshop.org